(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2007-524144
(P2007-524144A)
(43)公表日　平成19年8月23日(2007.8.23)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 9/50	(2006.01)	G06F	9/46	465A	5B042
G06F 11/30	(2006.01)	G06F	11/30	G	

審査請求 有　予備審査請求 未請求　（全 24 頁）

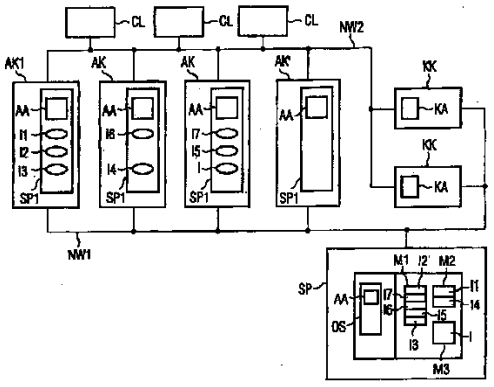
(21)出願番号	特願2006-515645 (P2006-515645)
(86)(22)出願日	平成16年3月16日 (2004.3.16)
(85)翻訳文提出日	平成17年12月19日 (2005.12.19)
(86)国際出願番号	PCT/DE2004/000530
(87)国際公開番号	WO2004/114143
(87)国際公開日	平成16年12月29日 (2004.12.29)
(31)優先権主張番号	10327601.7
(32)優先日	平成15年6月18日 (2003.6.18)
(33)優先権主張国	ドイツ(DE)
(31)優先権主張番号	10330322.7
(32)優先日	平成15年7月4日 (2003.7.4)
(33)優先権主張国	ドイツ(DE)

(71)出願人　505467959
フジツー　シーメンス　コンピュータース　ゲゼルシャフト　ミット　ベシュレンクテル　ハフツング
ドイツ連邦共和国　８０８０７　ミュンヘン　ドマークシュトラーセ　２８
(74)代理人　100075166
弁理士　山口　巖
(72)発明者　フリース、ベルンハルト
ドイツ連邦共和国　６８７６６　ホッケンハイム　イン　デア　クラム　２８
(72)発明者　ナヴァビ、グラーナ
ドイツ連邦共和国　７６６６９　バート　シェーンボルン　ツォイターナー　シュトラーセ　３５

最終頁に続く

(54)【発明の名称】クラスタ装置

(57)【要約】
【課題】少ない構成費用にて作動させることのできるクラスタ装置を提供する。
【解決手段】第１のネットワーク（ＮＷ１）と、アプリケーションノード（ＡＫ）を成しかつオペレーティングシステム（ＯＳ）を有する少なくとも２つのデータ処理システムと、各アプリケーションノード（ＡＫ）におけるアプリケーションエージェント（ＡＡ）とを有するクラスタ装置において、アプリケーションエージェント（ＡＡ）が次の機能を有する。すなわち、
－　アプリケーションノード（ＡＫ）において実行されるインスタンス（Ｉ１，Ｉ２，Ｉ３）の監視（ＵＢ）および認識、
－　新たなインスタンス（Ｉ３）の始動またはアプリケーションノードにおいて予定よりも早く終了させられたインスタンスの再始動、
－　アプリケーションノード（ＡＫ）における新たなインスタンス（Ｉ３）の実行が可能であるか否かの評価（ＢＥ）および判定、
－　ネットワーク（ＮＷ１）に接続されているアプ



【特許請求の範囲】
【請求項１】
－　第１のネットワーク（ＮＷ１）を備えていること、
－　それぞれ１つのアプリケーションノード（ＡＫ）を成しかつそれぞれ１つのオペレーティングシステム（ＯＳ）を有する少なくとも２つのデータ処理システムを備え、アプリケーションノード（ＡＫ）は実行されるインスタンス（Ｉ１，Ｉ２，Ｉ３，Ｉ４）を有すること、
－　各アプリケーションノード（ＡＫ）に、次の機能を有するアプリケーションエージェント（ＡＡ）を備えていること、すなわち、
　－　アプリケーションノード（ＡＫ）において実行されるインスタンス（Ｉ１，Ｉ２，Ｉ３）の監視（ＵＢ）および識別；
　－　新たなインスタンス（Ｉ３）の始動（ＳＴ）またはアプリケーションノードにおいて予定よりも早く終了させられたインスタンスの再始動；
　－　アプリケーションノード（ＡＫ）における新たなインスタンス（Ｉ３）の実行が可能であるか否かの評価（ＢＥ）および決定；
　－　ネットワーク（ＮＷ１）に接続されているアプリケーションノードのアプリケーションエージェント（ＡＡ）へのインスタンス実行要求（ＡＦ）；
　－　ネットワーク（ＮＷ１）に接続されているアプリケーションノード（ＡＫ）のアプリケーションエージェント（ＡＡ）へのインスタンス（Ｉ３）の実行要求（ＡＦ）の引き受け後のメッセージ通知（ＭＥ）；
を特徴とするクラスタ装置。
【請求項２】
　アプリケーションエージェントの監視（ＵＢ）の機能はリスト（Ｔ）の作成（Ｌ）を含み、リスト（Ｔ）は、それぞれアプリケーションノード（ＡＫ）において実行されるインスタンス（Ｉ１，Ｉ２）と、実行されるインスタンスの実行に必要な全てのデータ（Ｄ）とを部分リスト（ＴＩ１，ＴＩ２）として含んでいることを特徴とする請求項１記載のクラスタ装置。
【請求項３】
　アプリケーションエージェント（ＡＡ）の監視（ＵＢ）の機能は、アプリケーションノードにおいて実行されるインスタンスと他のインスタンスおよび／またはパラメータとの関連性を認識するように構成されていることを特徴とする請求項１又は２記載のクラスタ装置。
【請求項４】
　アプリケーションエージェント（ＡＡ）は、インスタンスの不安定な動作状態を認識するように構成されている機能を有することを特徴とする請求項１乃至３の１つに記載のクラスタ装置。
【請求項５】
　アプリケーションエージェント（ＡＡ）は、アプリケーションノードにおいて実行されるインスタンスを終了させるための機能を有することを特徴とする請求項１乃至４の１つに記載のクラスタ装置。
【請求項６】
　リスト（Ｔ）はアプリケーションノードに関する情報も有することを特徴とする請求項２記載のクラスタ装置。
【請求項７】
　記憶装置（ＳＰ）は、第１のネットワークに接続されていて、かつアプリケーションノード（ＡＫ）において実行可能な少なくとも１つのインスタンス（Ｉ１）を有することを特徴とする請求項１乃至６の１つに記載のクラスタ装置。
【請求項８】
　アプリケーションノードのアプリケーションエージェントは、アプリケーションノードにおいて動作するオペレーティングシステム（ＯＳ）のサービスであることを特徴とする

請求項１乃至７の１つに記載のクラスタ装置。
【請求項９】
記憶装置（ＳＰ）にはアプリケーションノード（ＡＫ）のためのオペレーティングシステム（ＯＳ）が格納されていることを特徴とする請求項７記載のクラスタ装置。
【請求項１０】
新たなインスタンス（Ｉ３）の実行が可能であるアプリケーションノード（ＡＫ’）が設けられていることを特徴とする請求項１乃至９の１つに記載のクラスタ装置。
【請求項１１】
クラスタ装置は第１のネットワークに接続された制御ノード（ＫＫ）として構成された少なくとも１つのデータ処理システムを有し、制御ノード（ＫＫ）がオペレーティングシステムおよび制御エージェント（ＫＡ）を有し、制御エージェント（ＫＡ）は、次の機能、すなわち、
－　第１のネットワーク（ＮＷ１）に接続されているアプリケーションノード（ＡＫ）の機能性の検査（ＵＰ）；
－　そのネットワーク（ＮＷ１）に接続されているアプリケーションノード（ＡＫ）のアプリケーションエージェント（ＡＡ）へのインスタンス実行要求（ＡＦ）；
－　アプリケーションノード（ＡＫ）の決定（ＢＳ）およびこのアプリケーションノードへの新たなインスタンスの実行要求；
を有することを特徴とする請求項１乃至１０の１つに記載のクラスタ装置。
【請求項１２】
アプリケーションノード（ＡＫ）の検査（ＵＰ）の際に、検査すべきアプリケーションノード（ＡＫ）は、アプリケーションノード（ＡＫ）のアプリケーションエージェント（ＡＡ）によって作成されたリスト（Ｔ）を介して求められ得ることを特徴とする請求項１１記載のクラスタ装置。
【請求項１３】
実行要求（ＡＦ）は、アプリケーションエージェント（ＡＡ）の作成された部分リスト（ＴＩ１，ＴＩ２）を有することを特徴とする請求項２又は１１記載のクラスタ装置。
【請求項１４】
作成されたリスト（ＴＩ１）および／または実行要求（ＡＦ）および／またはメッセージ（ＭＥ）は、少なくとも１つのファイルとして記憶装置（ＳＰ）に格納されていることを特徴とする請求項１乃至１３の１つに記載のクラスタ装置。
【請求項１５】
記憶装置（ＳＰ）に格納されている各インスタンスは、インスタンスが実行されるアプリケーションノード（ＡＫ）に割り当てられる一義的な識別番号を有することを特徴とする請求項１乃至１４の１つに記載のクラスタ装置。
【請求項１６】
第１のネットワーク（ＮＷ１）は、ＴＣＰ／ＩＰまたはＮＦＳプロトコルによる通信を行うように構成されていることを特徴とする請求項１乃至１５の１つに記載のクラスタ装置。
【請求項１７】
少なくとも１つの制御ノード（ＫＫ）はアプリケーションノード（ＡＫ）の初期化プロセスを制御するための手段を有し、初期化プロセスはアプリケーションノード（ＡＫ）のオペレーティングシステム（ＯＳ）の読み込みのためのコマンドを有することを特徴とする請求項１１乃至１６の１つに記載のクラスタ装置。
【請求項１８】
少なくとも２つのアプリケーションノード（ＡＫ，ＡＫ’）が第２のネットワーク（ＮＷ２）に接続され、第２のネットワーク（ＮＷ２）はクライアントコンピュータ（ＣＬ）に接続されていることを特徴とする請求項１乃至１７の１つに記載のクラスタ装置。
【請求項１９】
記憶装置（ＳＰ）に格納されているインスタンスはデータベースを有することを特徴と

する請求項１乃至１８の１つに記載のクラスタ装置。
【請求項２０】
少なくとも１つのデータ処理システムに識別番号（ＩＰ１，ＩＰ２）が割り当てられていることを特徴とする請求項１乃至１９の１つに記載のクラスタ装置。
【請求項２１】
クラスタ装置のデータ処理システム（ＤＶ）は同じ識別番号（ＩＰ１，ＩＰ２）によりプール（ＶＣ１，ＶＣ２，ＶＣ３）を成すことを特徴とする請求項２０記載のクラスタ装置。
【請求項２２】
アプリケーションノード（ＡＫ）のアプリケーションエージェント（ＡＡ）は、次の機能を有する、すなわち
－　アプリケーションノード（ＡＫ）に割り当てられた識別番号（ＩＰ１，ＩＰ２）の評価；
－　同じ識別番号（ＩＰ１，ＩＰ２）を有するアプリケーションノード（ＡＫ）におけるアプリケーションエージェント（ＡＡ）へのインスタンスの実行要求；
－　同じ識別番号（ＩＰ１，ＩＰ２）を有するアプリケーションノード（ＡＫ）におけるアプリケーションエージェント（ＡＡ）へのインスタンス実行要求の引き受け後のメッセージ通知；
を有することを特徴とする請求項２０乃至２１の１つに記載のクラスタ装置。
【請求項２３】
識別番号（ＩＰ１，ＩＰ２）はＩＰアドレスまたはＩＰアドレスの一部を含むことを特徴とする請求項２０乃至２２の１つに記載のクラスタ装置。
【請求項２４】
制御ノード（ＫＫ１）として構成されたデータ処理システム（ＤＶ）に識別番号（ＩＰ２）が割り当てられていて、制御ノード（ＫＫ１）において実行される制御エージェント（ＫＡ）は同じ識別番号を有するアプリケーションノード（ＡＫ）の機能を検査するように構成されていることを特徴とする請求項２０乃至２３の１つに記載のクラスタ装置。
【請求項２５】
アプリケーションノード（ＡＫ）において実行されるインスタンス（Ｉ１，Ｉ２，Ｌ１，Ｌ２，Ｌ３）に優先順位（ＰＳ１，ＰＳ２）が割り当てられていて、アプリケーションエージェント（ＡＡ）は、評価、判定およびこれらのインスタンスの実行要求という機能において、優先順位（ＰＳ１，ＰＳ２）を評価するように構成されていることを特徴とする請求項２０乃至２４の１つに記載のクラスタ装置。
【請求項２６】
制御ノード（ＫＫ１）の制御エージェント（ＫＡ）は、評価、判定およびこれらのインスタンスの実行要求の際にアプリケーションノード（ＡＫ）において実行されるインスタンスの優先順位（ＰＳ１，ＰＳ２）を評価するように構成されていることを特徴とする請求項２０乃至２５の１つに記載のクラスタ装置。
【請求項２７】
ネットワーク（ＮＷ１）に接続されている少なくとも２つのアプリケーションノード（ＡＫ，ＡＫ’）および制御ノード（ＫＫ）からなるクラスタ装置における方法であって、
－　制御ノード（ＫＫ）が、アプリケーションノード（ＡＫ，ＡＫ’）から、ノードにおいて実行される全てのインスタンス（Ｉ１，Ｉ２，Ｉ３）とインスタンスの実行のために必要なデータとを有するリスト（Ｔ）を受け取り、
－　制御ノード（ＫＫ）がアプリケーションノード（ＡＫ）の故障を規則的な時間間隔にて検査し、
－　制御ノード（ＫＫ）が、アプリケーションノード（ＡＫ，ＡＫ’）の故障時に、故障アプリケーションノードにおいて実行されるインスタンスと実行に必要なデータとを有するリスト（ＴＩ３）を作成して、ネットワーク（ＮＷ１）に接続されているアプリケーションノード（ＡＫ，ＡＫ’）に実行要求と共に転送することを特徴とする方法。

【請求項２８】
制御ノード（ＫＫ）が、アプリケーションノードの故障時に、アプリケーションノードによって作成されたリスト（ＴＩ３）を実行要求と一緒に少なくとも１つの他のアプリケーションノード（ＡＫ）に転送することを特徴とする請求項２７記載の方法。
【請求項２９】
アプリケーションノードがインスタンスの実行のためのリスト（Ｔ）を作成し、このリスト（Ｔ）と共に要求を少なくとも１つの他のアプリケーションノードに伝えることを特徴とする請求項２７記載の方法。
【請求項３０】
制御ノード（ＫＫ）が、アプリケーションノードの故障時に、アプリケーションノードによって作成されたリスト（Ｔ）と調整可能なパラメータとにより他のアプリケーションノードを確定し、これに故障アプリケーションノードにおいて実行されるインスタンスの実行要求を伝達することを特徴とする請求項２７又は２８記載の方法。
【請求項３１】
制御ノード（ＫＫ）が、検査すべきアプリケーションノードを、アプリケーションエージェントが受け取ったリスト（Ｔ）の評価によって確定することを特徴とする請求項２７乃至３０の１つに記載の方法。
【請求項３２】
方法が、制御ノードにおいて実行される制御エージェント（ＫＡ）およびアプリケーションノードにおいて実行されるアプリケーションエージェント（ＡＡ）によって実施されることを特徴とする請求項２７乃至３１の１つに記載の方法。
【請求項３３】
少なくとも１つのアプリケーションノード（ＡＫ）および制御ノード（ＫＫ）に識別番号（ＩＰ２）が割り当てられていて、
－　制御ノード（ＫＫ１）の制御エージェント（ＫＡ）が、同じ識別番号（ＩＰ２）を有する少なくとも１つのアプリケーションノード（ＡＫ）のアプリケーションエージェント（ＡＡ）からテスタメントを受け取ることを特徴とする請求項２７乃至３２の１つに記載の方法。
【請求項３４】
－　実行されるインスタンス（Ｉ１，Ｉ２，Ｌ１，Ｌ２，Ｌ３）に優先順位（ＰＳ１，ＰＳ２）が割り当てられ、
－　制御エージェント（ＫＡ）が、アプリケーションノード（ＡＫ）の故障時に、故障アプリケーションノード（ＡＫ）において実行されるインスタンスに割り当てられた優先順位（ＰＳ１，ＰＳ２）を評価し、優先順位（ＰＳ１，ＰＳ２）に依存して実行要求を送信することを特徴とする請求項１７乃至３０の１つに記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、ネットワークに接続されている少なくとも２つのアプリケーションノードおよび１つの制御ノードからなるクラスタ装置およびクラスタ装置における方法に関する。
【０００２】
ネットワークを介して結合されかつ設定された課題を共通に処理する複数のコンピュータからなる相互接続はクラスタと呼ばれる。処理すべき課題は小さな部分課題に分解され、個々のコンピュータに割り振られる。この種の公知のクラスタが、とりわけ非常に計算費用のかかる課題に使用されるＢｉｏｗｕｌｆクラスタ（ｗｗｗ．ｂｉｏｗｕｌｆ．ｏｒｇ）である。他の形式のクラスタの場合には、クラスタの計算速度ではなくて可用性が関心の的である。この形式のクラスタの場合には、クラスタ内の１つのコンピュータの故障時に他のコンピュータが故障コンピュータの課題を時間損失なしにまたは僅かな時間損失のみで引き継ぐことが保証されなければならない。このようなクラスタの例がインターネット内のウェブサーバであり、またはリレーショナルデータベースを有する中央記憶装置

でもある。

【０００３】

このような動作様式を有するクラスタは、高可用性クラスタとも呼ばれ、ネットワークを介して互いに接続されている複数の個別サーバを有する。各サーバはクラスタのノードを形成する。アプリケーションが処理されるサーバはアプリケーションノードと呼ばれ、中央の管理、制御または監視の課題を有するサーバは制御ノードを形成する。アプリケーションノードでは種々のアプリケーションまたは大きなアプリケーションの種々の部分アプリケーションが実行され、個々のアプリケーションは互いに関連することができる。クライアントと呼ばれるクラスタ外のコンピュータは、クラスタ内で実行されるアプリケーションにアクセスしてデータを呼び出す。

【０００４】

このようなクラスタは、アプリケーションノードのほかに、中央インスタンスである制御ノードを含む。制御ノードは、個々のアプリケーションノードにおける実行中のアプリケーションを監視し、場合によってはこれを終了させ、またはそれを新たに始動させる。１つのアプリケーションノードの故障時には、中央インスタンスが残りのアプリケーションノード上において、中止されたアプリケーションを新たに始動させる。このために制御ノードは、なおも十分に容量を有するノードを選択する。その際に、クラスタの構成および稼働率に応じて、今まで使用されていなかったアプリケーションノードが使用され、または新たに始動されるアプリケーションの計算負荷ができるだけ一様に配分される。この過程が負荷バランシングと呼ばれる。

【０００５】

他方では中央インスタンスまたは制御ノードを故障から守るために、大抵は中央インスタンスの役目をする他のサーバによって中央インスタンスに冗長性を持たせることが必要である。しかしながら、このようなクラスタ解決策は、アプリケーションノードと中央インスタンスとの間におけるデータ交換が非常に大きくなるという欠点を有する。これに加えて、各アプリケーションノードは中央インスタンスの照会に応答するために計算時間を消費する。更に、制御ノードはあらゆる有り得る故障シナリオを処理することができなければならないために、構成費用およびそれにつながる欠陥のある構成のリスクが少なからず増大する。

【０００６】

本発明の課題は、明白に少ない構成費用にて作動させることのできるクラスタ装置を提供することにある。

【０００７】

本発明は並列関係にある特許請求の範囲によって解決される。

【０００８】

本発明による装置においては、第１のネットワークと、それぞれ１つのアプリケーションノードを成しかつそれぞれ１つのオペレーティングシステムを有する少なくとも２つのデータ処理システムとを備えたクラスタが設けられている。各アプリケーションノードはアプリケーションエージェントを有し、少なくとも１つのアプリケーションノードは実行されるインスタスを有する。アプリケーションエージェントは少なくとも次の機能を含んでいる。すなわち、

－　アプリケーションノードにおいて実行される全てのインスタンスの機能性および誤りのない動作の監視、

－　新たなインスタンスの自立した始動またはアプリケーションノードにおいて予定よりも早く終了したインスタンスの再始動、

－　アプリケーションノードにおける新たなインスタンスの実行が可能であるか否かの評価および決定、

－　ネットワークに接続されているアプリケーションノードのアプリケーションエージェントへのインスタンス実行要求、

－　ネットワークに接続されているアプリケーションノードのアプリケーションエージェ

ントへのインスタンス実行要求の引き受け後のメッセージ通知。

【０００９】

各アプリケーションノードには、自立的にかつ他のアプリケーションノードにおけるアプリケーションエージェントならびに中央インスタンスに依存せずに動作するアプリケーションエージェントまたはプログラムが設けられている。特に、アプリケーションエージェントまたはプログラムは、新たなインスタンスの始動もしくは実行が可能であるか否かに関する自立した決定到達ができるように構成されている。アプリケーションエージェントは、アプリケーションエージェントがインスタンス実行要求を受け取るか、またはインスタンスの実行が誤りのない動作のために必要になるときに、この機能を実行する。評価は記憶、要求に含まれる情報および予め定義された規則を含めて行なうことが望ましい。アプリケーションエージェントの記憶は、それが過去の動作状態に関する情報を含むように構成されている。

【００１０】

付加的に、アプリケーションエージェントは他のアプリケーションエージェントへの要求のための機能を有する。それによって、エージェントには、差し迫った故障の際に、自立して中央インスタンスに依存せずに、アプリケーションノードにおいて実行されるインスタンスを他のアプリケーションノードに引き渡すことが可能である。代替として、この機能により、インスタンス実行要求を他のアプリケーションエージェントに送信することもできる。この機能は、実行すべきインスタンスの誤りのない動作のための全てのパラメータを準備するという趣旨で実施されている。

【００１１】

特に、アプリケーションエージェントはこれらの機能により、アプリケーションノードに限定された自立した負荷バランシングを行なうことができる。

【００１２】

各アプリケーションノードにおけるアプリケーションエージェントの待機により中央インスタンスは負担を軽減され、中央インスタンスとアプリケーションノードとの間におけるデータ交信が低減され、監視機能がアプリケーションノードへ移される。アプリケーションエージェントはそれらの側において独立している。

【００１３】

アプリケーションエージェントがインスタンス実行要求を拒絶するとき、他のアプリケーションノードのアプリケーションエージェントへのメッセージを発生する機能を有することが望ましい。

【００１４】

アプリケーションノードにおけるアプリケーションエージェントの監視機能がリスト作成を含む場合に特に有利である。リストは、それぞれアプリケーションノードにおいて実行されるインスタンスと、実行されるインスタンスの実行に必要な全てのデータおよびパラメータを含んでいる。望ましくは、リストが部分リストに分割されていて、各部分リストが１つのインスタンスのためのデータを含んでいる。更に、これらのリストがアプリケーションノードに関する情報および動作パラメータも含んでいると有利である。これらは可能な実行に関する評価の際に使用されることが好ましい。特に簡単なやり方では、ネットワークに接続されているアプリケーションノードのアプリケーションエージェントへのリストもしくは部分リストの発送による要求が行なわれる。

【００１５】

本発明による発展形態では、クラスタ装置は第１のネットワークに接続されている記憶装置を有する。記憶装置はアプリケーションノードのアクセスのために構成されている。記憶装置は、アプリケーションノードにおいて実行可能な少なくとも１つのインスタンスを含んでいる。それによって、クラスタ装置のアプリケーションノードは、記憶装置内のインスタンスにアクセスして実行のために取り込むことができる。全てのデータを記憶装置に保存し、これらを全てのアプリケーションノードに利用可能にすることが望ましい。この共通利用される記憶装置はコストを低減し、クラスタ装置のメンテナンスを簡単化す

る。

【００１６】

これに関連して、記憶装置にアプリケーションノードのためのオペレーティングシステムが格納されているとよい。それによって、オペレーティングシステムは各アプリケーションノードに個々にインストールされなくてもよく、アプリケーションノードの初期化過程において記憶装置から読み込まれる。したがって、オペレーティングシステムにおける更新および変更を間単に実施することができる。各アプリケーションノードのアプリケーションエージェントがアプリケーションノードで作動するオペレーティングシステムのサービスであることが望ましい。アプリケーションエージェントは共通利用される記憶装置に格納されている。アプリケーションエージェントがアプリケーションノードの初期化の際に自動的に始動されると有意義である。

【００１７】

クラスタ装置の実施形態においては、新たなインスタンスの実行が可能であるアプリケーションノードが設けられている。したがって、クラスタ装置は、アプリケーションノードの故障時に、故障ノードにおいて動作中のアプリケーションを引き継ぐアプリケーションノードを常に含んでいる。

【００１８】

有利な発展形態においては、クラスタ装置は制御ノードとして構成された少なくとも１つのデータ処理システムを有し、データ処理システムは第１のネットワークに接続されている。制御ノードはオペレーティングシステムおよび制御エージェントを有し、制御エージェントは、次の機能を有する。すなわち、

－　第１のネットワークに接続されているアプリケーションノードの機能性の検査、

－　そのネットワークに接続されているアプリケーションノードのアプリケーションエージェントへのインスタンス実行要求、

－　アプリケーションノードの決定およびこのアプリケーションノードへの新たなインスタンスの実行要求。

【００１９】

制御ノードにおけるこのような制御エージェントを用いることにより、アプリケーションノードと制御ノードとの間のデータ交換が明白に低減される。特に、アプリケーションノードの機能性の検査を簡単な周期的な存在テストによって行なうことができる。合理的な構成においては、アプリケーションノードのアプリケーションエージェントがそれの存在および機能性を制御エージェントによって検査される。個々のインスタンスの検査は省略される。なぜならば、これはアプリケーションエージェントによって行なわれるからである。アプリケーションノードの全体的な故障の際には制御エージェントが故障インスタンスの実行のための新たなアプリケーションノードを決定する。それによって常に誤りのない動作が保証される。

【００２０】

アプリケーションノードの検査の際に、検査すべきアプリケーションノードを、アプリケーションノードのアプリケーションエージェントによって作成されたリストを介して求めることができる。その際にアプリケーションノードのアプリケーションエージェントによって作成されたリストを制御エージェントが任意に使用することができ、制御エージェントがこれを評価する。リストによって制御エージェントはクラスタ装置内に存在するアプリケーションノードに関する知識を得る。したがって、クラスタ装置内に存在するノードの動的な探索が不要となる。特に、本発明による装置のこの構成においては、クラスタ装置に更なるアプリケーションノードを間単に追加することができる。制御エージェントへのリストの伝達後に、新たなアプリケーションノードの周期的な検査が行なわれる。

【００２１】

インスタンスの実行のための要求は、アプリケーションノードのアプリケーションエージェントの作成された部分リストを有することが好ましい。実行要求の機能は、制御エージェントにおいて、またアプリケーションエージェントにおいて、等しく構成されている

ことが望ましい。

【００２２】

本発明の望ましい発展形態においては、アプリケーションエージェントによって作成されたリストおよび／または実行要求および／または実行要求引き受けメッセージが記憶装置の少なくとも１つのファイルとして格納されている。これは、中央インスタンスもしくは制御インスタンスの存在なしに、各アプリケーションエージェントによるアクセスおよび独立した評価を可能にする。更に、要求が記憶装置のメモリ領域におけるリストの簡単な準備によって通知され、このリストの除去によって引き受け後の通報がなされることが好ましい。

【００２３】

本発明の発展形態では、記憶装置に格納された各インスタンスが一義的な識別番号を有し、識別番号はインスタンスが実行されるアプリケーションノードに割り当てられる。それによって、各アプリケーションノードにおける各インスタンスを既に実行された他のインスタンスに依存せずに実行させることができる。第２のインスタンスへの第１のインスタンスのアクセスは一義的な識別番号を介して行なわれる。それによって、個々のインスタンスおよびアプリケーションエージェントはクラスタ装置の構造的な構成に関する知識を必要としない。１つのノードにおいて複数のインスタンスが実行される場合には、複数の識別番号を割り当てることも勿論可能である。一実施形態においては一義的な識別番号は仮想のＩＰアドレスである。

【００２４】

第１のネットワークはＴＣＰ／ＩＰまたはＮＦＳプロトコルによる通信を行うように構成されているとよい。両プロトコルは多数のデータ伝送および管理の可能性をもたらし、特に簡単に実現することができる。

【００２５】

これに関連して、少なくとも１つの制御ノードはアプリケーションノードの初期化プロセスを制御するための手段を有し、初期化プロセスはアプリケーションノードのオペレーティングシステムの読み込みのためのコマンドを有することが望ましい。したがって、この手段により、制御ノードによってアプリケーションノードにおける初期化プロセスが始動され、初期化プロセスがオペレーティングシステムのロードをもたらす。特に、初期化プロセスの間にアプリケーションノードのためのコマンドもしくはパラメータが引き渡される。

【００２６】

本発明の発展形態では、クラスタ装置における少なくとも２つのアプリケーションノードおよび少なくとも１つの制御ノードが第２のネットワークに接続され、第２のネットワークがクライアントコンピュータに接続されている。このコンピュータは、大抵はアプリケーションノードにおいて実行されるインスタンスへの照会を送信するために使用される。第２のネットワークの構成によって、アプリケーションノード間のデータ流と、クライアントコンピュータとアプリケーションノードとの間のデータ流とが分離される。したがって、１つのネットワークにおける交換データ量が低減され、同時に分離によって監視または不当なアクセスに対する信頼性が高められる。

【００２７】

特別に有利な構成は次の構成である。すなわち、記憶装置に格納されているインスタンスがデータベースの一部として構成されていることである。代替的には、格納されたインスタンスはデータベースにアクセスするアプリケーションである。この場合にデータベースは記憶装置の一部であることが好ましい。

【００２８】

本発明の他の発展形態においては、各データ処理システムに識別番号が割り当てられている。同じ識別番号を有するクラスタ装置のデータ処理システムはプールを成す。したがって、簡単なやり方でクラスタ装置が更に分割されて、個別の課題を引き受けることができる。

【００２９】
ネットワークに接続されている少なくとも２つのアプリケーションノードおよび制御ノードからなるクラスタ装置における方法は、制御ノードが、アプリケーションノードからアプリケーションノードにおいて実行される全てのインスタンスとインスタンスの実行のために必要なデータおよびパラメータとを有するリストを受け取ることを特徴とする。更に、制御ノードがアプリケーションノードを故障について規則的な時間間隔にて検査し、アプリケーションノードの故障時に、故障アプリケーションノードにおいて実行されるインスタンスと実行に必要なデータとを有するリストを作成する。制御ノードは、このリストを、ネットワーに接続されているアプリケーションノードに実行要求と共に転送する。
【００３０】
この方法により、制御ノードはアプリケーションノードの故障を、換言するならば、アプリケーションノードの存在を検査するだけである。アプリケーションノードにおいて実行されるインスタンスの検査、管理または監視は放棄される。それによってアプリケーションノードと制御ノードとの間のデータ量が明白に低減される。本方法の発展形態では、アプリケーションノードが、ノードで実行されるインスタンスの状態変化時に、変化通知または新しいリストを送信する。
【００３１】
制御ノードは、監視機能によって記録されたアプリケーションノードの故障の際に、アプリケーションノードによって作成されたリストを実行要求と一緒に少なくとも１つの他のアプリケーションノードに転送する。代替として、アプリケーションノードが、実施のために決定されたインスタンスと実施のために必要なデータとを有するリストを作成し、このリストを少なくとも１つの他のアプリケーションノードに伝える。
【００３２】
他の構成においては、制御ノードが、アプリケーションノードの故障時に、アプリケーションノードによって作成されたリストと調整可能なパラメータとにより他のアプリケーションノードを求める。求められたアプリケーションノードには、故障アプリケーションノードにおいて実行されるインスタンスの実行要求が送信される。それによって、効率的なやり方にて負荷バランシングがクラスタ全体のための制御ノードにより行なわれる。これに関連して、制御ノードが、第１のアプリケーションノードに対しては、インスタンス終了のための信号を送信し、第２のアプリケーションノードに対しては、その終了させられたインスタンスを実行させるための信号を送信する。
【００３３】
特に、作成されたリストおよび調整可能なパラメータにより、実行を予定よりも早く終了させられたインスタンスの実行のために適切なコンピュータを見つけ出すことができる。更に、制御ノードが、検査すべきアプリケーションノードを、アプリケーションノードによって得られたリストを評価することによって求めるとよい。一構成例では本方法は、制御ノードにおいて実行される制御エージェントおよびアプリケーションノードにおいて実行されるアプリケーションエージェントによって実施される。
【００３４】
他の有利な構成が従属請求項からもたらされる。更に、図面を参照しながら実施例に基づいて本発明を詳細に説明する。図１はクラスタ装置の第１の実施例を示し、図２は使用される概念を説明するためのダイアグラムを示し、図３はアプリケーションエージェントの機能概要を示し、図４は制御エージェントを示し、図５は図１による本発明装置の一部を示し、図６はアプリケーションエージェントおよび制御エージェントの一動作態様の実施例を示し、図７はプール形成を有するクラスタ装置の第２の実施例を示し、図８は読み書き可能なメモリ領域の一部を概略的に示し、図９はアプリケーションエージェントによって管理されるテスタメントの一部を概略的に示す。
【００３５】
図１は６つのデータ処理システムを有する本発明によるクラスタ装置を示す。サーバとして構成されているこれらのデータ処理システムのうちの４つは、アプリケーションノー

ドＡＫ１，ＡＫ，ＡＫおよびＡＫ’を成す。他の２つのサーバはそれぞれ１つの制御ノードＫＫを成す。全てのサーバは主プロセッサならびにメインメモリおよび／またはハードディスク記憶装置ＳＰ１を有する。各アプリケーションノードもしくは制御ノードのメモリはオペレーティングシステムＯＳを有する。オペレーティングシステムＯＳは、アプリケーションノードにおけるプログラム制御のための機能、動作監視および維持のための機能、そしてノードの個々の構成要素へのアクセスのための機能を有する。更に、アプリケーションノードＡＫ１，ＡＫおよびＡＫ’のメモリＳＰ１は、それぞれ、オペレーティングシステムＯＳの一部であるアプリケーションエージェントＡＡを有する。制御ノードＫＫのメモリは制御エージェントＫＡを含む。

【００３６】

各アプリケーションノードＡＫ１，ＡＫおよびＡＫ’は第１のネットワークＮＷ１を介して制御ノードＫＫならびに記憶装置ＳＰに接続されている。このネットワークは、アプリケーションノードＡＫ１，ＡＫおよびＡＫ’間相互のデータ転送、アプリケーションノードＡＫ１，ＡＫおよびＡＫ’と制御ノードＫＫとの間のデータ転送、そしてアプリケーションノード、制御ノードおよび記憶装置ＳＰ間のデータ転送を可能にする。第２のネットワークＮＷ２はアプリケーションノードおよび制御ノードをクライアントコンピュータＣＬに接続する。クライアントコンピュータはアプリケーションノードに照会するために構成されていて、照会は処理のためにアプリケーションノードに送られる。

【００３７】

記憶装置ＳＰは全てのアプリケーションノードＡＫ１，ＡＫおよびＡＫ’のためのオペレーティングシステムＯＳを有する。各アプリケーションノードのアプリケーションエージェントＡＡはこのオペレーティングシステムＯＳのサービスであり、オペレーティングシステムの初期化後に始動される。それはバックグランドで動作するデーモンである。更に、記憶装置ＳＰは多数のプログラムモジュールＭ１，Ｍ２，Ｍ３を含む。これは、他方では個々のインスタンスに分割することのできる大きなアプリケーションである。例えばモジュールＭ１は５つのインスタンスを有し、モジュールＭ２は２つのインスタンスを有し、そしてモジュールＭ３は１つのインスタンスＩからなる。

【００３８】

種々のモジュールのインスタンスがアプリケーションノードＡＫ１およびＡＫのメモリＳＰ１にロードされていて、そこで実行される。例えばアプリケーションノードＡＫ１はモジュールＭ２のインスタンスＩ１ならびにモジュールＭ１のインスタンスＩ２およびＩ３を実行し、両アプリケーションノードＡＫはインスタンスＩ４～Ｉ７ならびにインスタンスＩを実行する。アプリケーションＡＫ’では他のインスタンスは実行されない。

【００３９】

モジュールとインスタンスとの間の関係は図２から読み取ることができる。モジュールＭは複数のコンピュータで実行される比較的大きなアプリケーションである。このためにモジュールＭはインスタンスと呼ばれる複数の小さな単位に分割されている。個々のインスタンスは、その必要が生じれば、互いに連絡し合ってデータを交換する。それによって関連性が生じる。他方では、インスタンスは個々のプロセスＰｒからなるサブインスタンスＳｕＩに分割されている。個々のインスタンスはそれのサブインスタンスＳｕＩおよびプロセスＰｒと一緒に１つのコンピュータ上で実行される。この場合にノードは異なるモジュールまたは同じモジュールに属する複数のインスタンスを実行することもできる。例えば、インスタンスＩ１およびＩ４を有するモジュールＭ２がアプリケーションＡＫ１およびＡＫに分配されている。ノードＡＫ１上では同時にモジュールＭ１のインスタンスＩ２およびＩ３が実行される。

【００４０】

各インスタンスには一義的な識別番号ＩＰ１，ＩＰ２およびＩＰ３が割り当てられていて、アプリケーションノードにおいてインスタンスを実行する際には、そのアプリケーションノードにこの識別番号が割り振られる。インスタンスが実行されるアプリケーションノードは、この一義的な識別番号ＩＰ１，ＩＰ２およびＩＰ３を確認することができる。

したがって、あるアプリケーションノードから他のアプリケーションノードへのインスタンスの交替が問題なく可能である。なぜならば、識別番号が古いアプリケーションノードでは抹消され、新たなアプリケーションノードに割り振られるからである。インスタンスへのアクセスは該当する一義的な識別番号を用いて行なわれる。本実施例では一義的な識別番号は仮想的なＩＰアドレスによって定義されている。したがって、アプリケーションノードＡＫ１は、インスタンスの仮想的なＩＰアドレスＩＰ１，ＩＰ２およびＩＰ３を受け取る。ネットワークＮＷ２を介してインスタンスＩ１にアクセスしようとするクライアントＣＬは、インスタンスＩ１に割り付けられている仮想的なＩＰアドレスに照会を送信する。アプリケーションノードＡＫ１はこの照会を受信してインスタンスＩ１に転送し、それをインスタンスＩ１が処理する。インスタンスに割り付けられていてそのインスタンスの実行時にアプリケーションノードに割り当てられる仮想的なアドレスの使用は、アプリケーションノードの自由な選択を可能にする。

【００４１】

アプリケーションエージェントＡＡの種々の機能が図３においてアプリケーションノードＡＫ１の例で説明されている。このアプリケーションノードではインスタンスＩ１，Ｉ２およびＩ３が実行される。アプリケーションエージェントＡＡは監視手段ＵＢを有し、それによりインスタンスを監視する。これに属するのが、例えば各インスタンスのための使用メモリのプロセッサ稼働率の測定、処理された照会および他の動作パラメータである。更に、アプリケーションエージェントＡＡは正しい動作態様およびインスタンスＩ１およびＩ３の可用性を監視する。更に、アプリケーションエージェントは、監視手段により、アプリケーションノードにおいて他の監視すべきインスタンスが存在するかどうかを検査する。監視手段は、監視すべきであるインスタンスを認識するように構成されている。これに加えて、監視手段は、アプリケーションノードにおいて実行されるインスタンス間の関連性を認識する。さらに、監視手段はとりわけノード上で実行中の全てのプロセスのリストを周期的に分析する。周期的な検査によって後から始動されたインスタンスも認識されて自動監視に引き継がれる。

【００４２】

更に、アプリケーションエージェントは、以後の経過においてテスタメントと呼ばれるリストＴの発生Ｌのための機能を有する。このテスタメントＴは、個別の監視すべきインスタンスＩ１～Ｉ３の全ての重要なデータＤが割り付けられた個々の部分テスタメントＴＩ１～ＴＩ３からなる。割り付けられたデータＤには、インスタンスの名称のほかにインスタンスの誤りのない動作のために必要な動作パラメータも属している。これのための例が、必要なメモリおよび計算容量、環境変数、他のインスタンスおよび動作パラメータに対するインスタンス同士の関連性などである。付加的にテスタメントにはアプリケーションノードに関するデータおよびパラメータが含まれている。これらは、例えば使用されるサーバの型および種類、名称、位置、メモリおよびプロセッサである。全てのアプリケーションノードのテスタメントのこれらのパラメータの評価がクラスタ構造の決定を可能にし、かつ他の設定可能性を与える。

【００４３】

監視機能が監視すべきインスタンスを見つけ出さない場合に、エージェントは、アプリケーションノードが新たなインスタンスの実行のために使用されることを認識し、このことを相応にテスタメントにおいてはっきり示す。

【００４４】

アプリケーションエージェントＡＡは始動機能ＳＴにより構成されている。したがって、アプリケーションエージェントＡＡは、記憶装置ＳＰからネットワークＮＷ１を介してインスタンスを取り込み、これをアプリケーションノード上で実行する。実行されるインスタンスは監視手段ＵＢによって周期的に検査される。

【００４５】

評価および判定機能ＢＥによりアプリケーションエージェントＡＡは新たなインスタンスが実行可能であるかどうかを評価し、アプリケーションエージェントＡＡはインスタン

スが始動されるべきであるアプリケーションノードを的確にとらえる。アプリケーションノードの動作パラメータ（プロセッサおよびメモリ稼働率）の測定および内部の記憶（テスタメントＴはこの記憶の一部である。）のほかに、新たなインスタンスの始動に関する判定は定められた規則に依存する。条件が満たされたならばアプリケーションエージェントＡＡはインスタンスを取り込んでこれを実行する。

【００４６】

定められた規則の例は、例えばプロセッサ能力およびメモリについての最小限準備の条件である。他の規則は、特定のインスタンスを定められた時間の間のみ実行するという定義付けである。ここでも他の規則は、新たなインスタンスの始動時に監視機能によりインスタンスの関連性を調べ、このインスタンスに関連したこれまで始動されていないインスタンスを同様に実行に至らしめることを意味する。

【００４７】

新たなインスタンスの始動後にアプリケーションエージェントはメッセージＭＥをネットワークＮＷ１を介して別のアプリケーションノードの他のアプリケーションエージェントに送信する。これにより、新たなインスタンスの始動が成功したことを示す。

【００４８】

アプリケーションエージェントは、個々のインスタンスＩ１，Ｉ２およびＩ３のための監視手段ＵＢによって、監視されるインスタンスＩ３の不慮の予定より早い終了を認識することができる。更に、誤りのない動作を維持するために、アプリケーションエージェントは障害のあるインスタンスＩ３の終了および再始動のための機能を有する。再始動が成功しなかった場合に、エージェントは、テスタメントＴから、障害のあるインスタンスＩ３の部分テスタメントＴＩ３を発生させ、部分テスタメントＴＩ３を有するこのインスタンスの始動のための要求ＡＦをネットワークＮＷ１を介して他のアプリケーションノードに送信する。その際にエージェントは障害のあるインスタンスＩ３の終了または更なる始動試行の停止を行なうことができる。

【００４９】

更に、アプリケーションエージェントはアプリケーションノードにおいて実行されるインスタンスを終了させる機能を有する。この機能はインスタンス終了要求にしたがって使用される。それによってインスタンスが1つのアプリケーションノードにおいて終了させられ、他のアプリケーションノードにおいて新たに実行される。

【００５０】

アプリケーションノードＡＫにおけるアプリケーションエージェントＡＡの独立かつ自立した判定到達によって、制御ノードまたは中央で動作する監視手段によるアプリケーションノードＡＫにおける個々のインスタンスの連続的な制御および監視はもはや必要でない。

【００５１】

図４は制御エージェントＫＡの機能に関する概要を示す。制御エージェントＫＡはアプリケーションエージェントＡＡからそれぞれのテスタメントを受け取り、これらを管理する。これによりアプリケーションエージェントがクラスタに登録される。制御エージェントは、アプリケーションエージェントＡＡのテスタメントＴから、クラスタ内に存在する全てのアプリケーションノードＡＫをそれらのハードウェア情報を含めて備えたリストを発生する。それによって、制御エージェントは自立的にクラスタの現在の構成情報を入手しかつ動的な変化も登録する。更に、制御エージェントＫＡは、ネットワークＮＷ１を介する全てのアプリケーションノードＡＫの機能性および存在の検査のための手段ＵＰを備えている。アプリケーションノードＡＫの機能性および存在は、アプリケーションエージェントからの簡単な存在信号の送信によって伝達される。例えば、制御エージェントＫＡがネットワークＮＷ１を介してＰｉｎｇ信号を個々のアプリケーションノードＡＫに送信することができる。

【００５２】

機能テストにおける応答不在によって知らされるアプリケーションノード障害の際には

、制御エージェントＫＡは当該アプリケーションノードＡＫに対するテスタメントを評価し、それから部分テスタメントＴＩ３を抽出する。この部分テスタメントはこのインスタンスの実行のための要求ＡＦと共にネットワークＮＷ１に導かれ、そして残っているアプリケーションノードＡＫに導かれる。これの代替として、制御エージェントＫＡは、インスタンスを実行するアプリケーションノードを決定するための機能を持っている。図１のクラスタ装置においてアプリケーションノードＡＫ’はインスタンスを持っていないので、制御エージェントＫＡは、アプリケーションノードＡＫ１の障害後におけるインスタンスＩ１，Ｉ２およびＩ３の実行のために、このアプリケーションノードＡＫ’を決定する。アプリケーションノードＡＫ１のアプリケーションエージェントＡＡから伝達されるテスタメントＴによって、中止されたインスタンスがアプリケーションノードＡＫ’において始動可能となる。

【００５３】

図５は新しいアプリケーションノードＡＫ”が付け加えられた本発明によるクラスタ装置の部分図を示す。アプリケーションノードＡＫでは２つのインスタンスＩ１およびＩ２が実行される。アプリケーションノードＡＫ’ではインスタンスは全く実行されない。アプリケーションノードＡＫのアプリケーションエージェントＡＡは、両部分テスタメントＴＩ１およびＴＩ２を有するテスタメントＴを作成し、これを制御ノードＫＫおよび制御エージェントＫＡに伝達したところである。ノードＡＫ’のアプリケーションエージェントは空のテスタメントＴ’を制御エージェントＫＡに伝達し、それのテスタメントＴ’における登録ＳＰによりアプリケーションノードＡＫ’がインスタンスの始動のために準備完了であることを知らせる。この登録によりノードＡＫ’が自由なノードとして明らかにされる。

【００５４】

制御エージェントＫＡは、自身の側で、アプリケーションノードＡＫおよびＡＫ’のテスタメントＴおよびＴ’を有するリストを管理する。エージェントＫＡは、周期的にノードのアプリケーションエージェントＡＡの状態信号を要求することによって、ノードの存在を検査する。アプリケーションノードＡＫにおける監視されるインスタンスＩの動作パラメータの変化、すなわち終了または新たなインスタンスの始動の際には、この変化がそれぞれのアプリケーションエージェントＡＡによって自動的に制御ノードＫＫの制御エージェントＫＡに伝達される。したがって、制御エージェントＫＡのリストは、常にアプリケーションノードＡＫのテスタメントの現在状態を含んでいる。更に、制御エージェントはアプリケーションノードのハードウェアパラメータに関する情報を受け取る。

【００５５】

ここで、新しいアプリケーションノードＡＫ”がネットワークＮＷ１に接続される。初期化段階後にアプリケーションエージェントＡＡがノードＡＫ”において始動する。エージェントＡＡの監視機能ＵＢが、アプリケーションＡＫ”において実行されるプロセス、インスタンスおよびアプリケーションを検査し、自動的にアプリケーションエージェントＡＡによって監視すべきインスタンスＩ３を認識する。アプリケーションノードの動作パラメータと一緒に、エージェントはそれからインスタンスＩ３の動作に必要な全てのデータおよびパラメータを含んだ部分テスタメントを有するテスタメントＴ”を発生する。アプリケーションノードＡＫ”の発生させられたテスタメントＴ”は制御エージェントＫＡに伝達される。それによりアプリケーションエージェントがクラスタにおいて登録され、アプリケーションノードＡＫ”においてインスタンスが実行され、そのインスタンスが監視される。制御エージェントＫＡは今やアプリケーションノードＡＫ，ＡＫ’およびＡＫ”の存在をそれのリストにあるテスタメントにしたがって検査する。

【００５６】

アプリケーションノードＡＫ”がネットワークＮＷ１から分離されるか、または予定より早く例えば電源障害によって遮断された場合には、存在の検査が否定的結果をもたらす。制御エージェントＫＡはテスタメントにあるインスタンスの実行要求を有するテスタメントＴ”をアプリケーションノードＡＫおよびＡＫ’に送信する。アプリケーションエー

ジェントＡＡはそのテスタメントを受信し、測定、記憶および外部パラメータにより、全体のテスタメントまたは部分テスタメントがアプリケーションノードにおいて実行可能であるか否かの判定が的確にとらえられる。

【００５７】

アプリケーションノードＡＫ’のアプリケーションエージェントは肯定的判定を的確にとらえて全体のテスタメントＴ”を受け取る。エージェントはインスタンスＩ３をテスタメントにおいて予め与えられたパラメータにしたがってそれのノードにて新たに始動し、今や新たなインスタンスＩ３の部分テスタメントを含む新しいテスタメントＴ’を制御エージェントに伝達する。

【００５８】

図６は他の好ましい構成を示す。クラスタ装置は２つのアプリケーションノードＡＫおよびＡＫ’と記憶装置ＳＰと制御ノードＫＫとを有し、これらはネットワークＮＷ１を介して互いに接続されている。ノード相互および記憶装置ＳＰとの通信はＴＣＰ／ＩＰプロトコルにより行なわれる。

【００５９】

実行されかつアプリケーションエージェントによって監視されるインスタンスＩ１，Ｉ３およびＩ２は記憶装置ＳＰに保存されているモジュールＭ２を成す。更に記憶装置ＳＰは、モジュールＭ１と、アプリケーションノードに共通に使用されるオペレーティングシステムＯＳとを含み、オペレーティングシステムはアプリケーションエージェントＡＡを有する。記憶装置ＳＰは、２つの部分領域Ｂ１およびＢ２に分割されている領域Ｂを有する。

【００６０】

領域Ｂは、全てのアプリケーションエージェントＡＡおよび制御エージェントＫＡのために読み書き可能に構成されている。アプリケーションエージェントは、それらのアプリケーションノードのテスタメントを記憶装置ＳＰにおける部分領域Ｂ１に保存する。アプリケーションノードにおける変化の際には、このノードのアプリケーションエージェントが新たなテスタメントを発生し、それにより領域Ｂ１において古いテスタメントを交換する。制御ノードの制御エージェントは領域Ｂ１におけるテスタメントを評価し、それにともなって監視すべきアプリケーションノードのリストを発生する。

【００６１】

更に、各ノードのアプリケーションエージェントは、記憶装置の領域Ｂ２を周期的に評価する。領域Ｂ２にはインスタンスの実行要求が保存されている。この構成においては、要求は領域Ｂ２へのテスタメントまたは部分テスタメントの格納によって行なわれる。アプリケーションエージェントは、領域Ｂ２に格納されているテスタメントまたは部分テスタメントを読み取って、実行に関する独立した判定を的確にとらえる。アプリケーションノードがテスタメントを引き受けることができる場合には、アプリケーションエージェントはそのテスタメントを領域Ｂ２から消去し、指定されたインスタンスを始動する。要求または引き受け後の通知は、領域Ｂ２へのテスタメントの格納またはその領域からのテスタメントの消去によって簡単なやり方にて行なわれる。要求の拒否はテスタメントが領域Ｂ２に残されていることによって自動的にもたらされる。

【００６２】

インスタンスを終了させるアプリケーションエージェントはそのテスタメントを領域Ｂ２に格納するので、他のアプリケーションノードがこれを引き受けることができる。アプリケーションノードが完全に故障し、アプリケーションエージェントがこれを前もって領域Ｂ２へのテスタメントの格納によって指定することができない場合には、制御エージェントが故障したアプリケーションノードのテスタメントを領域Ｂ２へ移動する。残りのノードのアプリケーションエージェントはそれらの側で判定を的確にとらえる。このやり方で高い柔軟性が達成される。多数の部分テスタメントへのテスタメントの分割によって、障害のあるアプリケーションノードのインスタンスを複数のノードに配分することができる。アプリケーションエージェントの独立および装置ＳＰの共通使用されるメモリによっ

て、制御ノードＫＫの障害発生時にも誤りのない動作が保証されている。

【００６３】

ここに挙げた例のほかに、多数の他の構成を見いだすことができる。特にアプリケーションエージェントの判定到達のための規則、制御エージェントおよびアプリケーションエージェントの機能および課題、そしてテスタメントにおけるパラメータが拡張可能である。

【００６４】

本発明の他の観点は、いわゆるクラスタ装置内の個別データ処理システムの仮想クラスタへのグループ化に関する。この場合に、クラスタ装置内の幾つかのデータ処理システムに同一識別番号が割り当てられ、そのようにしてこれらのデータ処理システムが１つのプールに統合される。ここにおいて、仮想という概念は、異なるデータ処理システム相互における規則によって定められた論理的な関連性にすぎない。複数のデータ処理システムによるクラスタ装置内のプール形成は、異なるアプリケーションを高可用性に保とうとする場合に特に有利である。幾つかのデータ処理システムを特別にデータベースサービスの実行のために設け、これに対して同じ物理的なクラスタの他のデータ処理システムはウェブアプリケーションのために設けるることが望ましい。

【００６５】

プール形成は一般的な規則にしたがって行なわれる。これらの規則は、例えば高可用性のアプリケーションに対する規則に関連し得るが、しかし例えば純粋なハードウェアパラメータも含み得る。更に、物理的なクラスタ内でのプール形成により、異なるユーザグループの個別プールを割り振ることが可能である。それぞれのユーザグループによって始動されたアプリケーションは、それぞれのプールに割り当てられたデータ処理システムにおいてのみ実行されて、高可用性に保たれる。相互に割り当てられている複数のデータ処理システムからなるこのようなプールは、物理的なクラスタ装置内の仮想クラスタとも呼ばれる。

【００６６】

図７は、その中に含まれている複数の仮想クラスタもしくはプールを有するクラスタ装置の実施形態を示す。そこに示されたクラスタは、共通のネットワークＮＷ１を介して複数のデータ処理システムに接続されている複数の記憶装置ＳＰ，ＳＰ１およびＳＰ２を含んでいる。これらのデータ処理システムはそれぞれコンピュータとして主プロセッサおよび主メモリを装備している。これらのデータ処理システムのうち１２個がアプリケーションノードＡＫ１～ＡＫ１２として構成されている。他の２つのデータ処理システムが制御ノードＫＫおよびＫＫ１を成す。アプリケーションノードＡＫ１～ＡＫ１２ではそれぞれ１つのアプリケーションエージェントＡＡが実行される。制御ノードＫＫおよびＫＫ１はそれぞれ１つの制御エージェントＫＡを含んでいる。

【００６７】

物理的なクラスタ装置は、この実施例において、３つの仮想クラスタＶＣ１，ＶＣ２およびＶＣ３を含む。仮想クラスタＶＣ１は、３つのアプリケーションノードＡＫ１０，ＡＫ１１およびＡＫ１２と、記憶装置ＳＰ１とを含んでいる。仮想クラスタＶＣ２は、アプリケーションノードＡＫ１～Ｋ５と、制御ノードＫＫ１と、記憶装置ＳＰ２とを含んでいる。更に、アプリケーションノードＡＫ３およびＡＫ４は、アプリケーションノードＡＫ６～ＡＫ９と一緒に仮想クラスタＶＣ３に割り当てられている。制御ノードＫＫおよび記憶装置ＳＰはこの実施例では仮想クラスタの一部ではない。

【００６８】

仮想クラスタのそれぞれのプールへの個々のアプリケーションノードＡＫもしくは制御ノードＫＫの割り振りは一般的な規則を介して行なわれる。これらの規則は、部分的には、外部のユーザによって予め与えられるか、制御ノードＫＫの制御エージェントＫＡによって予め与えられるか、またはアプリケーションノードＡＫにおける個々のアプリケーションエージェントのテスタメントからもたらされる。個々の仮想クラスタの同定のために、そして１つのプールへの物理的クラスタの個々のノードの割り振りのために、ＩＰアド

レスの一部分またはＩＰアドレス自体を使用するのが有利である。例えば、仮想クラスタＶＣ１のアプリケーションノードＡＫ１０，Ａｋ１１およびＡＫ１２には、部部的に一致するＩＰアドレスが割り付けられる。ＩＰアドレスが同じ部分を有する物理的クラスタ装置のノードは、同じプールもしくは仮想クラスタに属する。この仮想クラスタにおけるノードからまたはノードへの通信は同様にこの識別番号を含んでいる。データ処理システムにおけるエージェントの相応の評価によって他の識別番号を有する通信は無視される。

【００６９】

仮想クラスタＶＣ２のアプリケーションノードＡＫ１においては、アプリケーションエージェントＡＡならびにインスタンスＬ１およびインスタンスＩ１が実行される。アプリケーションノードＡＫ２はインスタンスＬ２およびＩ２を含む。アプリケーションノードＡＫ１のインスタンスＩ１およびアプリケーションノードＡＫ２のインスタンスＩ２は、」共通に１つのアプリケーションを成す。これらの共通なアプリケーションは、仮想クラスタＶＣ２の異なるアプリケーションノードＡＫ１～ＡＫ５において高可用性に保たれている。したがって、共通アプリケーションＩのための両インスタンスの一方Ｉ１もしくはＩ２の始動のための要求は、相応のアプリケーションノードが仮想クラスタＶＣ２に割り当てられているときにのみ引き受けられる。

【００７０】

アプリケーションノードＡＫ３は、アプリケーションエージェントＡＡのほかに部分インスタンスＬ３ならびにインスタンスＬ３１およびＬ３２も含み、これらはそれぞれ高可用性にて実行される。インスタンスＬ３はアプリケーションノードＡＫ２およびＡＫ１のインスタンスＬ２およびＬ１と一緒に仮想クラスタＶＣ２の他のアプリケーションを成す。アプリケーションノードＡＫ４およびＡＫ５は予備ノードであり、予備ノードにおいては仮想ノードＶＣ２の他のインスタンスは実行されない。

【００７１】

更に、アプリケーションノードＡＫ３およびＡＫ４は、仮想クラスタＶＣ３の構成要素でもある。したがって、要求の評価およびインスタンス始動要求の発送のために、アプリケーションノードＡＫ３およびＡＫ４におけるアプリケーションエージェントＡＡが相応の要求を常に同じプールに所属しているノードのアプリケーションエージェントに送信することが必要である。このために、例えばノードＡＫ３におけるアプリケーションエージェントＡＡが、仮想クラスタへの個々のインスタンスの一義的な割り当ての相関性に関して拡張されたテスタメントを既に含んでいる。

【００７２】

このテスタメントからの一部が図９において見ることができる。テスタメントは２つの比較的大きな部分領域に分けられていて、部分領域はそれぞれノードＡＫ３を構成部分とする両プールのアプリケーションを含む。これはプールもしくは仮想クラスタＶＣ２ならびにＶＣ３である。仮想クラスタＶＣ２は、仮想クラスタへの一義的割り当てを可能にする識別番号ＩＰ１を含む。更に、インスタンスＬ３が仮想クラスタＶＣ３において実行される。したがって、インスタンスＬ３に関係する全ての通知に対して対応関係ＩＰ１が一緒に送られる。ノードが同じ対応関係を持っていないところのエージェントは、この仮想クラスタの一部ではなく、したがって通知を無視する。

【００７３】

第２の部分領域は、仮想クラスタＶＣ３に割り当てられかつノードで実行されるインスタンスに対する全ての部分テスタメントを含む。プールＶＣ３は識別番号ＩＰ３を有する。したがって、テスタメントのこの部分領域内においては、アプリケーションエージェントＡＡによってインスタンスＬ３１およびＬ３２が管理される。それぞれの部分テスタメントＬ３１およびＬ３２は、これらの部分テスタメントを実行するためのパラメータのほかに、仮想クラスタＶＣ３への対応関係も含んでいる。インスタンスの１つの故障および不成功に終わった故障インスタンスの新たな初期化の際に、ノードＡＫ３のアプリケーションエージェントＡＡは、このインスタンスが実行される仮想クラスタに対する割り当て識別番号を有する部分インスタンスを発生する。インスタンスＬ３１およびＬ３２の障害

時には、仮想クラスタＶＣ３に識別番号ＩＰ３によって割り当てられている部分インスタンスが発生され、インスタンスＬ３の障害時には割り当て識別番号ＩＰ２を有する相応の部分インスタンスが発生される。

【００７４】

エージェントＡＡは、アプリケーションノードに、このインスタンスの実行要求を送信する。アプリケーションノードにおけるそれぞれのアプリケーションエージェントはこの要求を評価して、手始めにそれが同じ識別を有する仮想クラスタの部分であるか否かを検査する。それがインスタンスを実行させるべき仮想クラスタの部分でない場合には、実行要求は無視される。ほかの場合にはそれに必要な実行のためのリソースが使用可能であるか否かが検査される。

【００７５】

プールＶＣ２には付加的に、可用性および障害安全性を高めるために、制御エージェントＫＡを有する制御ノードＫＫ１が割り当てられている。このために、アプリケーションノードＡＫ１～ＡＫ５におけるアプリケーションエージェントＡＡが、それらの部分インスタンスを発生し、それらを大容量記憶装置ＳＰ２における共通な読み書き可能なメモリ領域に格納する。制御ノードＫＫ１における制御エージェントＫＡは、個々のアプリケーションノードＡＫ１およびそれらのエージェントの機能性を規則的な時間間隔で送出される状態メッセージによって監視する。

【００７６】

このメモリ領域およびアプリケーションノードＡＫ１～ＡＫ５におけるアプリケーションエージェントの個々の伝達されるテスタメントの概略図を図８に示す。全体リストにおいては、とりわけどのアプリケーションノードＡＫが仮想クラスタＶＣ２に割り当てられているかが整理されている。更に、仮想クラスタＶＣ２において目下のところ実行されるアプリケーションを有するリストが作成される。詳細には、これは部分インスタンスＩ１およびＩ２を有するアプリケーションならびに部分インスタンスＬ１，Ｌ２およびＬ３を有するアプリケーションである。これらのアプリケーションのそれぞれに優先順位が割り付けられている。それぞれの部分インスタンスはこれらの優先順位を受け継ぐ。優先順位は、仮想クラスタＶＣ２において実行される個々のアプリケーションがどの程度重要であるかを指定する。したがって、優先順位は実行されるアプリケーションの順序もしくはランク順を成す。

【００７７】

この例では、両部分インスタンスＩ１およびＩ２を有するアプリケーションが優先順位指標ＰＳ１を持ち、部分インスタンスＬ１，Ｌ２およびＬ３を有するアプリケーションが優先順位指標ＰＳ２を持つ。この場合に優先順位指標ＰＳ２は指標ＰＳ１よりも小さい。したがって、部分インスタンスＬ１，Ｌ２およびＬ３を有するアプリケーションが部分インスタンスＩ１およびＩ２を有するアプリケーションよりも重要でない。

【００７８】

更に、記憶装置ＳＰ２のメモリ領域は個々のアプリケーションノードＡＫ１～ＡＫ５のテスタメントＴ１～Ｔ５を含んでいる。これらは、それぞれのアプリケーションノードＡＫ１～ＡＫ５において作動する部分インスタンスのための部分テスタメントを含んでいる。アプリケーションノードＡＫ４およびＡＫ５のテスタメントＴ４およびＴ５は空である。

【００７９】

制御ノードＫＫの制御エージェントＫＡは、一般に個々のアプリケーションノードの高可用性を監視する。今、例えばアプリケーションノードＡＫ１が完全に故障した場合には、もはや部分インスタンスＩ１もしくはＬ１も実行されない。制御ノードＫＫにおける制御エージェントＫＡは、今やテスタメントＴ１から部分インスタンスＩ１およびＬ１のための２つの部分テスタメントを発生する。その際に高いほうの優先順位指標に基づいてインスタンスＩ１を有する部分テスタメントがこの部分インスタンスの実行要求と一緒に物理的なクラスタ内における個々のアプリケーションノードにネットワークを介して送信さ

れる。この部分テスタメント内において、実行すべきインスタンスＩ１がどの仮想クラスタに割り当てられているかが指定されている。

【００８０】

仮想クラスタＶＣ２に割り当てられていないアプリケーションノードＡＫにおけるアプリケーションエージェントＡＡは、実行要求を無視する。それに反してアプリケーションノードＡＫ２～ＡＫ５におけるエージェントＡＡはそれらのリソースを検査する。アプリケーションエージェントＡＡの１つが、場合によっては部分テスタメントを引き受け、インスタンスＩ１を自身のノードにおいて実行へと至らしめる。部分インスタンスＩ２の始動後に相応のメッセージが制御エージェントＫＡに返送される。部分インスタンスＩ１が部分テスタメントと一緒に仮想クラスタ内のアプリケーションエージェントＡＡの１つによって引き継がれて実行に成功したときにはじめて、制御エージェントＫＡが部分インスタンスＬ２を有する部分テスタメントを実行要求と一緒に送信する。

【００８１】

個々のアプリケーションもしくはインスタンスの優先順位付与によって、高い優先順位を有するアプリケーションが常に高可用性に保たれる。十分な容量が存在するときのみ、低い優先順位を有するインスタンスも新たに実行に至らしめられる。この実施例においては、仮想クラスタＶＣ２内の制御ノードＫＫ１における制御エージェントＫＡが、部分テスタメントの発生および部分インスタンスの実行要求を引き受ける。

【００８２】

アプリケーションノードＡＫ３の障害の見極めがつく場合には、これをノードＡＫ３のアプリケーションエージェントＡＡが確認する。このノードのエージェントＡＡは部分インスタンスＬ３を有する部分テスタメントおよび実行要求を発生し、これを物理的クラスタ装置および仮想クラスタ装置のアプリケーションノードにおけるエージェントに送信する。更に、アプリケーションノードＡＫ３におけるアプリケーションエージェントＡＡは、部分インスタンスＬ３１およびＬ３２を有する２つの部分テスタメントを発生し、これらを同様に実行要求と共にクラスタ装置に送信する。しかしながら、部分インスタンスＬ３１およびＬ３２は仮想クラスタＶＣ３に割り当てられていて、アプリケーションノードＡＫ１，ＡＫ２およびＡＫ５によって無視される。しかしながら、適切な自由なリソースにおいては、アプリケーションノードＡＫ４もしくはＡＫ６～ＡＫ９が部分インスタンスＬ３１およびＬ３２を引き受けることができる。

【００８３】

この実施例においては、メッセージが物理的なクラスタ装置内の全てのアプリケーションノードに送信される。しかしながら、メッセージが同じプールにおけるノードに由来する場合にのみ処理が行なわれる。拡張においては、同じ仮想クラスタ内のノードのみにメッセージを送信することもできる。それにより、確かにデータ発生が低減されるが、しかし柔軟性も縮小される。

【００８４】

加えて、アプリケーションノードにおいて計画的でなく終了させられる低い優先順位のインスタンスは、次の場合にはアプリケーションノード全体の再始動のためにアプリケーションエージェントを動かすことができないように配慮することが望ましい。すなわち、このノードにおいて、より高い優先順位を有するインスタンスがなおも誤りなく実行される場合である。例えば、アプリケーションノードＡＫ２におけるアプリケーションエージェントＡＡは、部分インスタンスＬ２の障害発生時に、より高い優先順位を有する部分インスタンスＩ２がなおも誤りなく実行される場合には、アプリケーションノードＡＫ２の全体の完全な再始動を行なわない。したがって、再始動のためには、アプリケーションエージェントＡＡが、部分インスタンスＩ２を有する部分テスタメントおよび実行要求を、仮想クラスタＶＣ２のアプリケーションノードに送信しなければならない。この部分インスタンスの引き受けおよび実行成功の確認時に、アプリケーションノードＡＫ２におけるアプリケーションエージェントＡＡは、アプリケーションノードＡＫ２の完全な再始動を初期化する。

【００８５】
部分ＩＰアドレスの助けによる個々の仮想クラスタへの割り当てによって、非常に動的にかつ柔軟にリソース要求における有り得る変化に応ずることができる。付加的に仮想クラスタ内においても個々のデータ処理システム間における更なるグループ化またはプール形成を設定することができる。仮想クラスタＶＣ３においては、例えばアプリケーションノードＡＫ８およびＡＫ９が仮想クラスタ内において他のグループを成す。このグループ化も一般的な規則を介して制御可能である。更に、完全なクラスタ構造を監視しかつ個々のデータ処理システムを規則的間隔にて存在を監視する制御エージェントＫＡを有する他の制御ノードＫＫが設けられるとよい。他のデータ処理システムが物理的クラスタに付け加えられるならば、この制御ノードは、仮想容量増大のために、付け加えられたデータ処理システムを異なる仮想クラスタに割り当てることができる。識別番号の割り付けによるプール形成の導入ならびにプール内の個々の部分インスタンスの優先順位付与は、物理的クラスタ装置内における非常に細かい段階付けおよび選択制御を可能にする。この場合に、個々のアプリケーションノードならびに物理的クラスタ装置内におけるプールが大幅に自動的に構成可能である。付加的に管理上の課題が大々的に解消する。

【００８６】
以上のとおり、制御ノードとアプリケーションノードとの間の明白に少ないデータ交換がこれと同時に大きな柔軟性をともなって達成される。個々のアプリケーションノードの自立によって完全に制御ノードを省略することさえも可能である。ノード経過の管理、制御および監視はアプリケーションエージェントの務めであり、これらのアプリケーションエージェントは互いに直接の通信を持っていない。アプリケーションエージェントの形成はインスタンスの独立の認識および監視を可能にする。それによって、高コストの構成を省略することができ、クラスタ構造が自立的に発生させられるためにクラスタ構造に関する正確な知識は必要でない。しばしばノード数を変更するクラスタを使用する場合にはまさに、独立した監視のこの構想は高い柔軟性をもたらす。

【図面の簡単な説明】

【００８７】
【図１】クラスタ装置の第１の実施例を示す概略図
【図２】使用される概念を説明するための概略図
【図３】アプリケーションエージェントの機能の概略図
【図４】制御エージェントを示す概略図
【図５】図１による本発明装置の部分詳細図
【図６】アプリケーションエージェントおよび制御エージェントの一動作態様の実施例を示す概略図
【図７】プール形成を有するクラスタ装置の第２の実施例を示す概略図
【図８】読み書き可能なメモリ領域の一部を示す概略図
【図９】アプリケーションエージェントによって管理されるテスタメントの一部を示す概略図。

【符号の説明】

【００８８】

ＡＫ，ＡＫ，ＡＫ’，ＡＫ”	アプリケーションノード
ＫＫ	制御ノード
ＫＡ	制御エージェント
ＡＡ	アプリケーションエージェント
ＣＬ	クライアントコンピュータ
ＮＷ１，ＮＷ２	ネットワーク
ＳＰ，ＳＰ１	記憶装置
Ｍ１，Ｍ２，Ｍ３	モジュール
Ｉ１，Ｉ２，・・・，Ｉ７	インスタンス
ＳＵＩ	副インスタンス

Ｐｒ	プロセス
ＩＰ１，ＩＰ２，ＩＰ３	識別番号
ＯＳ	オペレーティングシステム
Ｔ，Ｔ’，Ｔ”	テスタメント
Ｔ１，Ｔ２，Ｔ３	部分テスタメント
Ｄ	データ
Ｌ，ＵＢ，ＳＴ，ＭＥ，ＡＦ，ＢＥ	機能
ＵＰ，ＢＳ	機能
ＳＰ	識別番号
Ｂ，Ｂ１，Ｂ２	メモリ領域

【図１】

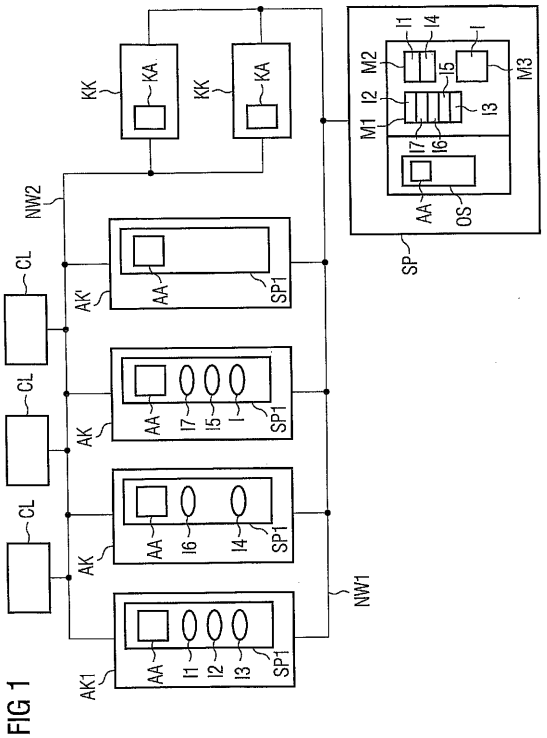
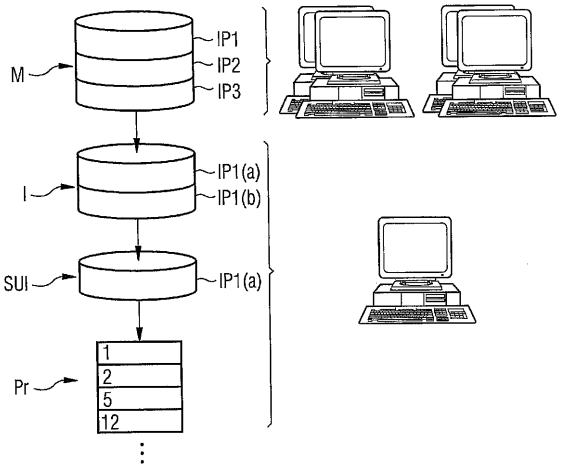


FIG 1

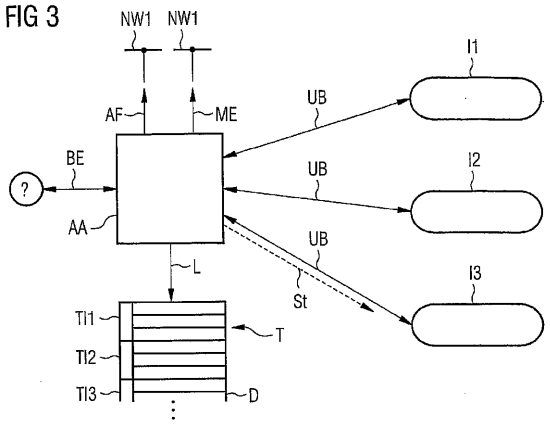
【図２】

FIG 2



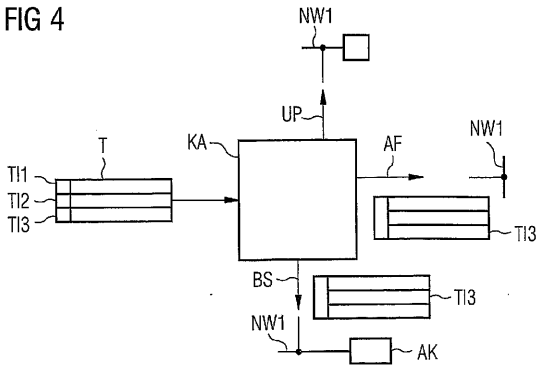
【図３】

FIG 3



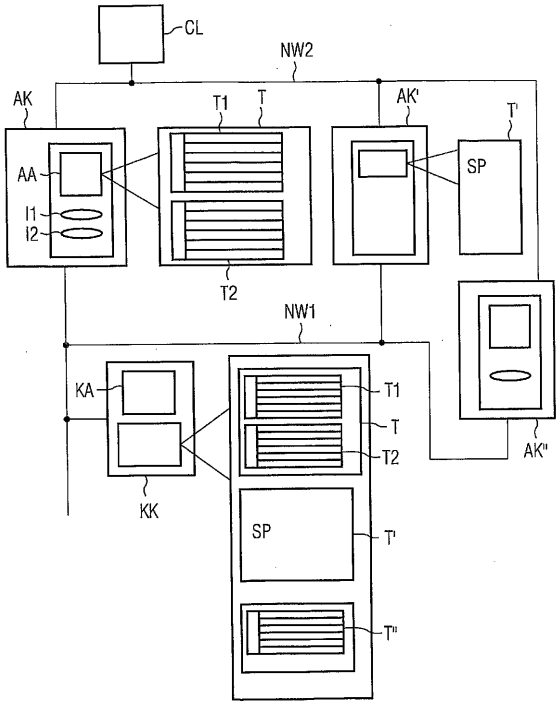
【図４】

FIG 4



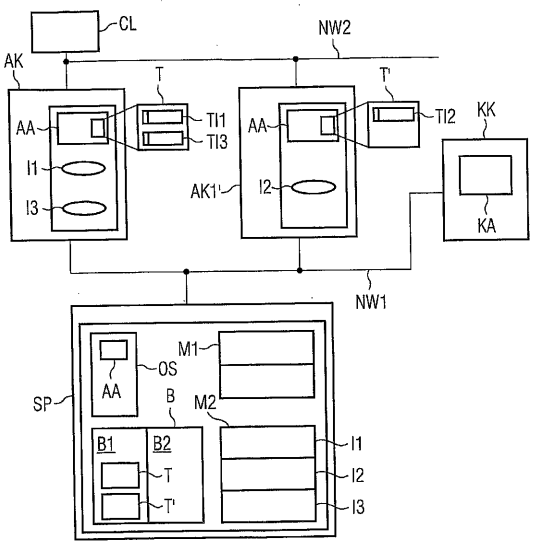
【図５】

FIG 5



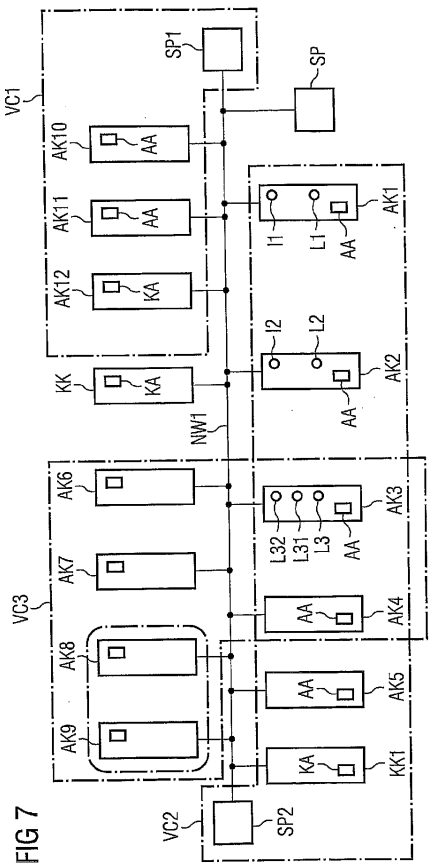
【図６】

FIG 6



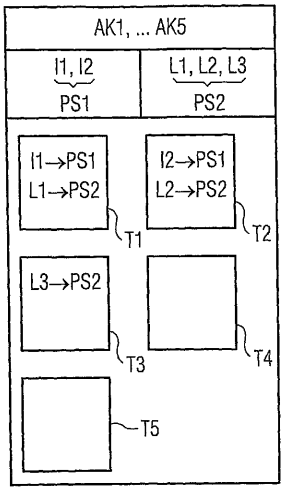
【図７】

FIG 7



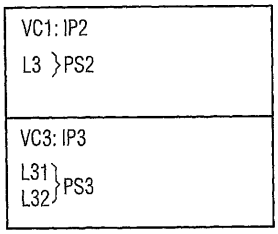
【図８】

FIG 8



【図９】

FIG 9



フロントページの続き

(81)指定国 AP(BW,GH,GM,KE,LS,MW,MZ,SD,SL,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HU,IE,IT,LU,MC,NL,PL,PT,RO,SE,SI,SK,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AT,AU,AZ,BA,BB,BG,BR,BW,BY,BZ,CA,CH,CN,CO,CR,CU,CZ,DK,DM,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,HR,HU,ID,IL,IN,IS,JP,KE,KG,KP,KR,KZ,LC,LK,LR,LS,LT,LU,LV,MA,MD,MG,MK,MN,MW,MX,MZ,NA,NI,NO,NZ,OM,PG,PH,PL,PT,RO,RU,SC,SD,SE,SG,SK,SL,SY,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,YU,ZA,ZM,ZW

(72)発明者 クラウス、ルディ
ドイツ連邦共和国 ６７１５８ エラーシュタット ポルトゥギーザーリング ５３

Ｆターム(参考) 5B042 GA12 GA19 GC10 GC18 JJ03 JJ05 KK17

【要約の続き】
リケーションノードのアプリケーションエージェント（ＡＡ）へのインスタンス実行要求（ＡＦ）、
－ ネットワーク（ＮＷ１）に接続されているアプリケーションノードのアプリケーションエージェント（ＡＡ）へのインスタンス実行要求（ＡＦ）引き受け後のメッセージ通知（ＭＥ）。